C. AMENDMENTS TO THE CLAIMS

Please amend the claims as follows:

1. (Currently amended) A method for programming a multithreaded application ~~so as~~ to minimize thread switching overheads and memory usage during processing of the multithreaded application, the method comprising:

   a. writing a plurality of errands, the plurality of errands being functions performing specific tasks that collectively constitute ~~the entire~~ a thread functionality; and

   b. forming at least one itinerary ~~one or more itineraries~~ corresponding to ~~a~~ the thread, the at least one itinerary ~~itineraries~~ controlling the execution of the plurality of errands in the desired manner.

2. (Cancelled)

3. (Currently amended) The method as recited in claim 1 wherein forming the at least one itinerary comprises:

   a. storing information regarding the plurality of errands that need to be executed, the information being stored in the form of at least one function pointers to at least one of the plurality of errands ~~the actual errand functions~~, the at least one function pointers being stored in an errand function list;

b. storing the sequence in which the plurality of errands are to be executed, the sequence being the order in which the ~~errand~~ at least one function pointers are stored within the errand function list; and

c. storing data required for execution of the plurality of errands within the at least one itinerary, the data being stored in an errand data list.

4. (Currently amended) The method as recited in claim 3 wherein forming the at least one itinerary further comprises defining an errand state field, the errand state field capable of being modified by preemptive errands, the preemptive errands being functions that may cause the thread to temporarily stall during ~~further~~ execution, the value assigned to the errand state field at any point of thread execution being indicative of the state of the thread.

5. (Currently amended) A method for executing a multithreaded application, ~~the multithreaded application having been programmed using itinerarized threads,~~ the multithreaded application comprising a plurality of threads, the plurality of threads comprising a plurality of standard threads and a plurality of itinerarized threads, the plurality of itinerarized threads comprising standard thread constructs and ~~as well as~~ at least one itinerary ~~itineraries~~, the at least one itinerary ~~itineraries~~ being lists of a plurality of errands, the plurality of errands being functions performing specific tasks, the method comprising:

a. compiling the multithreaded application ~~code~~;

b. scheduling the plurality of threads for execution;

c. executing the plurality of threads by running the standard thread constructs in a normal mode, the normal mode of execution being thread execution in accordance with a standard thread execution methodology, the normal [[-]]mode execution of a thread being carried out using an execution stack associated with the plurality of threads ~~thread~~;

d. executing the at least one itinerary ~~itineraries~~ of the plurality of itinerarized threads in an itinerary mode, the itinerary mode being a special thread execution scheme for executing ~~the~~ a complete itinerary, the itinerary mode execution of ~~a~~ at least one of the plurality of itinerarized threads being carried out using a kernel stack; and

e. exiting the itinerary mode of execution when the complete itinerary corresponding to ~~a~~ the at least one of the plurality of itinerarized threads has been executed, the at least one of the plurality of itinerarized threads being subsequently executed in normal mode.

6. (Currently amended) The method as recited in claim 5 wherein the itinerary mode execution is carried out after preempting the at least one of the plurality of itinerarized threads running in normal mode in response to a request for running the at least one itinerary on behalf of the at least one of the plurality of itinerarized threads.

7. Cancelled)

8. (Currently amended) The method as recited in claim 5 ~~7~~ wherein executing the plurality of threads in the normal mode comprises:

   a. loading ~~the~~ a thread context, the thread context being information of one of the plurality of threads required for execution of the one of the plurality of threads ~~thread~~;

   b. executing thread specific logic; and

   c. preempting the one of the plurality of threads ~~thread~~ in response to a request for ~~the~~ thread preemption, the thread preemption being done after storing the thread context in the execution stack associated with the one of the plurality of threads ~~thread~~ and storing a pointer to the execution stack in the thread structure.

9. (Currently amended) The method as recited in claim 5 wherein executing the at least one itinerary of the plurality of itinerarized threads in the itinerary mode comprises:

   a. setting up the at least one itinerary, the setting up of the at least one itinerary comprising:

      i. setting up the kernel stack, the kernel stack being set up in such a manner that it becomes the execution stack for the at least one itinerary;

      ii. setting up an errand state variable, the errand state variable being subsequently used for ascertaining the state of the ~~thread running the~~ plurality of errands ~~errand~~;

iii. setting up an errand function list, the errand function list storing at least one function pointers to the plurality of errands ~~actual errand functions~~;

iv. setting up an errand data list, the errand data list storing at least one data pointer pointing to data required for execution of the plurality of errands; and

v. storing at least one function pointers corresponding to the errand function list and at least one data pointer corresponding to the errand data list, the pointers being used subsequently for accessing the errand function list and the errand data lists; and

b. executing the plurality of errands corresponding to the at least one itinerary sequentially, the sequence of the plurality of errands being specified through the errand function list, the plurality of errands being executed using the data stored in the errand data list.

10.(Currently amended) The method as recited in claim 9 wherein the sequence of executing the plurality of errands is modified by conditional errands, the conditional errands being errands changing ~~that change~~ the sequence of errand execution based ~~upon certain~~ a predefined criteria ~~conditions~~.

11.(Currently amended) The method as recited in claim 9 wherein executing the plurality of errands sequentially comprises:

while the complete itinerary is not executed, iteratively performing the following steps:

a. loading ~~an~~ at least one of the plurality of errands for execution, the errand being loaded in accordance with the ~~specified sequence of errands in the~~ errand function list;

b. running ~~the~~ at least one of the plurality of errands and receiving a ~~the~~ return value returned by the at least one of the plurality of errands, a true return value indicating that the at least one of the plurality of errands was executed successfully, a false return value indicating that the at least one of the plurality of errands blocked and did not complete successfully;

c. running ~~executing the next~~ another of the plurality of errands in the errand function list in response to a true return value returned by the at least one of the plurality of errands ~~being executed~~;

d. blocking the at least one itinerary in response to a false return value returned by the at least one of the plurality of errands ~~being executed~~, the blocking being done in the itinerary mode; and

e. resuming ~~itinerary~~ execution of the at least itinerary when the blocked itinerary is scheduled back, the ~~itinerary~~ execution of the at least itinerary being resumed from the at least one of the plurality of errands that had caused the at least itinerary to block.

12.(Original) The method as recited in claim 11 wherein running an errand comprises:

a. checking the value of the errand state field of the itinerary, the errand state field being checked for ascertaining whether the errand is being executed for the first

time or it is a previously blocked errand, the errand returning a true value if it is a previously blocked errand;

b. executing the complete errand in case the errand is being executed for the first time;

c. returning a true return value if the complete errand is executed successfully; and

d. modifying the errand state field and returning a false return value if the errand is blocked, the errand state field being modified to indicate that the errand is being blocked.

13.(Currently amended) A multithreaded application processing system executing a multithreaded application ~~program~~, the multithreaded application ~~program~~ comprising a plurality of ~~standard threads and itinerarized~~ threads, the plurality of threads comprising a plurality of standard threads and a plurality of itinerarized threads, the plurality of standard threads having been written using standard thread methodology, the plurality of itinerarized threads comprising standard thread constructs and at least one itinerary ~~itineraries~~, the at least one itinerary ~~itineraries~~ being lists of a plurality of errands, the plurality of errands being functions performing specific tasks, the system comprising:

a. a compiler compiling the multithreaded application ~~program~~;

b. a memory storing information related to the plurality of threads, the memory comprising:

i. a plurality of thread stacks, each of the plurality of thread stacks being associated with at least one of the plurality of threads, the plurality of thread stack being stored with context information pertaining to the plurality of threads ~~thread~~, the context information being the information set required for processing of the plurality of threads ~~thread~~; and

ii. a kernel stack, the kernel stack being used by the plurality of itinerarized threads while the at least one itinerary is ~~itineraries are~~ being processed;

c. a processor processing the plurality of threads, the processor accessing the memory for information pertaining to the plurality of threads; and

d. an operating system scheduling and managing execution of the plurality of threads, the operating system executing the plurality of standard threads and standard thread constructs of the plurality of itinerarized threads in accordance with a standard thread execution methodology, the operating system executing the at least one itinerary ~~itineraries~~ in an itinerary mode, the itinerary mode being a thread execution scheme for executing ~~the~~ a complete itinerary.

14.(Currently amended) The system as recited in claim 13 wherein the multithreaded application processing system utilizes multiple processors, each of the multiple processors having a separate kernel stack associated with it in the memory.

15.(Currently amended) The system as recited in claim 13 wherein the operating system comprises:

a. a scheduler scheduling the plurality of ~~various standard and itinerarized~~ threads for execution on the basis of a pre-defined criteria;

b. a standard thread running service executing the plurality of ~~various~~ standard threads and standard thread constructs of the plurality of itinerarized threads; and

c. an itinerary running service executing the at least one itinerary ~~itineraries~~ corresponding to the plurality of itinerarized threads in the itinerary mode.

16.(Cancelled)

17. (Currently amended) The system as recited in claim 15 ~~16~~ wherein the scheduler maintains separate ready queues for holding the plurality of standard threads and the plurality of itinerarized threads.

18.(Currently amended) The system as recited in claim 15 wherein the standard thread running service executing the plurality of standard threads and standard thread constructs of the plurality of itinerarized threads comprises:

a. a preemption service enabling thread preemption;

b. a standard preemptive services, the preemptive services being specific activities that might preempt at least one of the plurality of itinerarized threads, the standard preemptive services using the preemption service for execution; and

c. a standard non-preemptive services, the non-preemptive services being specific activities that do not ~~need~~ preempt at least one of the plurality of itinerarized threads ~~to preempt~~.

19.(Currently amended) The system as recited in claim 15 wherein the itinerary running service comprises:

a. a preemption service enabling thread preemption;

b. a standard preemptive services, the preemptive services being specific activities that can preempt ~~the~~ at least one of the plurality of itinerarized threads, the standard preemptive services using the preemption service for execution;

c. a standard non-preemptive services, the non-preemptive services being specific activities that do preempt ~~not need a~~ at least one of the plurality of itinerarized threads ~~to preempt~~; and

d. an itinerary building service, the itinerary building service facilitating execution of itinerary specific program constructs.

20.(Currently amended) An itinerary running service for executing ~~an~~ a plurality of itinerarized threads in itinerary mode, the plurality of itinerarized threads comprising standard thread constructs and at least one itinerary ~~itineraries~~, the at least one itinerary ~~itineraries~~ being ~~lists~~ of a plurality of errands, the plurality of errands being small tasks needed to be performed during thread execution, the itinerary mode being a special thread execution scheme for executing ~~the~~ a complete itinerary, the itinerary running service receiving the at least one itinerary from an operating system scheduler, the itinerary running service comprising:

a. a preemptive services ~~means for~~ providing various preemptive ~~and non-preemptive~~ services to the at least one itinerary, the preemptive services being

specific activities that may preempt at least one of the plurality of itinerarized threads ~~the thread to preempt~~ [[,]] ~~the non-preemptive services being specific activities that do not need the thread to preempt~~;

b. a non-preemption services providing various non-preemptive services to the at least one itinerary, the non-preemptive services being specific activities that do not preempt the at least one of the plurality of itinerarized thread; and

c. an itinerary building service providing pre-programmed standard errands and enabling specification of application specific errands, the pre-programmed standard errands being various preemptive and non-preemptive services provided by the system, the application specific errands being special program constructs built into the at least one itinerary, further the itinerary building service executing the plurality of errands in the at least one itinerary sequentially and blocking the at least one itinerary when one of the plurality of errands blocks. [[;]]

~~d. means for executing the errands in the itinerary sequentially; and~~

~~e. means for blocking the itinerary when one of the errands blocks.~~

21.(Currently amended) The itinerary running service as recited in claim 20 further ~~comprising means for~~ resuming execution of a previously blocked itinerary from the errand that blocked the itinerary, when the itinerary is scheduled back on the itinerary running service by the operating system scheduler.

22.(Currently amended) A computer program product for executing a multithreaded application~~, the application having been programmed using itinerarized threads~~, the

multithreaded application comprising a plurality of threads, the plurality of threads comprising a plurality of standard threads and a plurality of itinerarized threads, the plurality of itinerarized threads comprising standard thread constructs ~~as well as~~ and at least one itinerary ~~itineraries~~, the at least one itinerary ~~itineraries~~ being lists of a plurality of errands, the plurality of errands being functions performing specific tasks, the computer program product comprising:

a computer readable medium comprising:

a. a program instruction ~~means~~ for compiling the multithreaded application ~~code~~;

b. a program instruction ~~means~~ for scheduling the plurality of threads for execution; and

c. a program instruction ~~means~~ for executing the plurality of threads in a manner that minimizes thread switching overheads and memory usage during the execution, the plurality of standard threads and the standard thread constructs of the plurality of itinerarized threads being executed using thread stacks associated with the threads as execution stacks, the at least one itinerary of the plurality of itinerarized threads ~~itineraries~~ being executed using kernel stack as their execution stack.

23.(Currently amended) The computer program product as recited in claim 22 wherein the computer readable medium comprising program instruction ~~means~~ for executing the plurality of itinerarized threads further comprises:

a. a program instruction ~~means~~ for executing threads running the standard thread constructs in a normal mode, the normal mode of execution being thread execution in accordance with standard thread execution methodology;

b. a program instruction ~~means~~ for preempting a thread running in the normal mode in response to a request for running ~~an~~ the at least one itinerary on behalf of ~~the~~ at least one of the plurality of itinerarized threads, the preempted thread subsequently being run in an itinerary mode, the itinerary mode being a special thread execution scheme for executing a ~~the~~ complete itinerary; and

c. program instruction ~~means~~ for exiting the itinerary mode of execution when the complete itinerary corresponding to ~~a~~ the preempted thread has been executed, the preempted thread being subsequently executed in the normal mode.